Samson March
Software Design for
Handheld Devices
Final Report

For my final project, I designed a heart diagnostic app. The idea behind the app was to use the microphone on an iOS device placed against the chest, neck or wrist of a patient and pick up any heart beats. The app itself unfortunately never fully came to fruition as the algorithm that I designed to pick up the heart beats and compare it with the library was too complex to convert (for my programming level) to objective-c. The algorithm which I created in MATLAB using an ECG signal as a test subject is included in the zip incase you would like to look over it and see that work.

In the objective-c code I was able to manage to get a sound recorded and to play back which, again for what I knew of object oriented programming was at least for me a huge accomplishment. It took me a long time to get what I do have up and running. The code uses the AVFoundation framework and in doing so calls AVAudioRecorder and AVAudioPlayer.

My code is setup to check if the device is recording first, if it is then the button that toggles the record/stop is set to stop and the device is recording the file to an NSURL. If the device is not currently recording then the button is set to record.

After the button has been pressed and an audio signal is recorded the same part of the code prepares the device to playback the audio. The audio is then able to be played back by pressing the play button.

The algorithm that was going to be used (if I could have figured out how to convert the matlab code to objective-c) works in the following fashion. First the data string is loaded as an array. The first column of data is stored as the time values. Next the second column of data is stored as just the ECG voltage (in the case of this app it would have been in terms of the microphone voltage output). The mean is subtracted out of the data in order get rid of any constant voltage. Next, the signal is passed through a lowpass filter in order to get rid of any high frequency noise and other artifacts in the signal that would ruin the analysis.

After this the data is checked for any values over a threshold value (in terms of standard deviations from the mean). An array of zeroes and ones is then created, a one is placed in the index value that corresponds to every time the signal is above the threshold and zeroes everywhere else. Next the same thing is done with the derivative of the signal. Following this, the two arrays are compared to each other, creating a new array of ones and zeroes. Whenever the two arrays have a value of one in the same index value then the new array has is given a one, other than that it is filled with zeroes. This is so that the values found correspond to only sharp edges of peaks and not merely points when the signal drifts above the threshold. Next, the new array is used to calculate the interval times between heart beats. The index time of one of the ones minus the index time of the previous one is the interval length in the time scale.

Following this the power spectrum distribution is calculated in order to determine the frequency of time intervals. In other words, how often a particular beat pattern occurs. Lastly if a certain percentage of the total power is under a threshold then the signal is deemed to have low variability.